

40.1 BOUND—Check Array Index Against Bounds

Opcode	Instruction	Description
62 /r	BOUND <i>r16,m16&16</i>	Check if <i>r16</i> (array index) is within bounds specified by <i>m16&16</i>
62 /r	BOUND <i>r32,m32&32</i>	Check if <i>r32</i> (array index) is within bounds specified by <i>m16&16</i>

Description

Determines if the first operand (array index) is within the bounds of an array specified the second operand (bounds operand). The array index is a signed integer located in a register. The bounds operand is a memory location that contains a pair of signed doubleword-integers (when the operand-size attribute is 32) or a pair of signed word-integers (when the operand-size attribute is 16). The first doubleword (or word) is the lower bound of the array and the second doubleword (or word) is the upper bound of the array. The array index must be greater than or equal to the lower bound and less than or equal to the upper bound plus the operand size in bytes. If the index is not within bounds, a BOUND range exceeded exception (#BR) is signaled. (When a this exception is generated, the saved return instruction pointer points to the BOUND instruction.)

The bounds limit data structure (two words or doublewords containing the lower and upper limits of the array) is usually placed just before the array itself, making the limits addressable via a constant offset from the beginning of the array. Because the address of the array already will be present in a register, this practice avoids extra bus cycles to obtain the effective address of the array bounds.

Operation

```
IF (ArrayIndex < LowerBound OR ArrayIndex > (UpperBound + OperandSize/8))
  (* Below lower bound or above upper bound *)
  THEN
    #BR;
FI;
```

Flags Affected

None.

Protected Mode Exceptions

#BR	If the bounds test fails.
#UD	If second operand is not a memory location.
#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.

- #PF(fault-code) If a page fault occurs.
- #AC(0) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

- #BR If the bounds test fails.
- #UD If second operand is not a memory location.
- #GP If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
- #SS If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

- #BR If the bounds test fails.
- #UD If second operand is not a memory location.
- #GP(0) If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
- #SS(0) If a memory operand effective address is outside the SS segment limit.
- #PF(fault-code) If a page fault occurs.
- #AC(0) If alignment checking is enabled and an unaligned memory reference is made.

40.2 BSF—Bit Scan Forward

Opcode	Instruction	Description
0F BC	BSF <i>r16,r/m16</i>	Bit scan forward on <i>r/m16</i>
0F BC	BSF <i>r32,r/m32</i>	Bit scan forward on <i>r/m32</i>

Description

Searches the source operand (second operand) for the least significant set bit (1 bit). If a least significant 1 bit is found, its bit index is stored in the destination operand (first operand). The source operand can be a register or a memory location; the destination operand is a register. The bit index is an unsigned offset from bit 0 of the source operand. If the contents source operand are 0, the contents of the destination operand is undefined.

Operation

```

IF SRC = 0
  THEN
    ZF ← 1;
    DEST is undefined;
  ELSE
    ZF ← 0;
    temp ← 0;
    WHILE Bit(SRC, temp) = 0
    DO
      temp ← temp + 1;
      DEST ← temp;
    OD;

```

FI:

Flags Affected

The ZF flag is set to 1 if all the source operand is 0; otherwise, the ZF flag is cleared. The CF, OF, SF, AF, and PF, flags are undefined.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit. If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

40.3 BSR—Bit Scan Reverse

Opcode	Instruction	Description
0F BD	BSR <i>r16,r/m16</i>	Bit scan reverse on <i>r/m16</i>
0F BD	BSR <i>r32,r/m32</i>	Bit scan reverse on <i>r/m32</i>

Description

Searches the source operand (second operand) for the most significant set bit (1 bit). If a most significant 1 bit is found, its bit index is stored in the destination operand (first operand). The source operand can be a register or a memory location; the destination operand is a register. The bit index is an unsigned offset from bit 0 of the source operand. If the contents source operand are 0, the contents of the destination operand is undefined.

Operation

```

IF SRC = 0
  THEN
    ZF ← 1;
    DEST is undefined;
  ELSE
    ZF ← 0;
    temp ← OperandSize - 1;
    WHILE Bit(SRC, temp) = 0
    DO
      temp ← temp - 1;
      DEST ← temp;
    OD;
FI;

```

Flags Affected

The ZF flag is set to 1 if all the source operand is 0; otherwise, the ZF flag is cleared. The CF, OF, SF, AF, and PF, flags are undefined.

Protected Mode Exceptions

- #GP(0) If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
If the DS, ES, FS, or GS register contains a null segment selector.
- #SS(0) If a memory operand effective address is outside the SS segment limit.
- #PF(fault-code) If a page fault occurs.
- #AC(0) If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

- #GP If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
- #SS If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

- #GP(0) If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
- #SS(0) If a memory operand effective address is outside the SS segment limit.
- #PF(fault-code) If a page fault occurs.
- #AC(0) If alignment checking is enabled and an unaligned memory reference is made.

40.4 BSWAP—Byte Swap

Opcode	Instruction	Description
0F C8+ <i>rd</i>	BSWAP <i>r32</i>	Reverses the byte order of a 32-bit register.

Description

Reverses the byte order of a 32-bit (destination) register: bits 0 through 7 are swapped with bits 24 through 31, and bits 8 through 15 are swapped with bits 16 through 23. This instruction is provided for converting little-endian values to big-endian format and vice versa.

To swap bytes in a word value (16-bit register), use the XCHG instruction. When the BSWAP instruction references a 16-bit register, the result is undefined.

Intel Architecture Compatibility

The BSWAP instruction is not supported on Intel Architecture processors earlier than the Intel486 processor family. For compatibility with this instruction, include functionally equivalent code for execution on Intel processors earlier than the Intel486 processor family.

Operation

```
TEMP ← DEST
DEST(7..0) ← TEMP(31..24)
DEST(15..8) ← TEMP(23..16)
DEST(23..16) ← TEMP(15..8)
DEST(31..24) ← TEMP(7..0)
```

Flags Affected

None.

Exceptions (All Operating Modes)

None.

40.5 BT—Bit Test

Opcode	Instruction	Description
0F A3	BT <i>r/m16,r16</i>	Store selected bit in CF flag
0F A3	BT <i>r/m32,r32</i>	Store selected bit in CF flag
0F BA /4 <i>ib</i>	BT <i>r/m16,imm8</i>	Store selected bit in CF flag
0F BA /4 <i>ib</i>	BT <i>r/m32,imm8</i>	Store selected bit in CF flag

Description

Selects the bit in a bit string (specified with the first operand, called the bit base) at the bit-position designated by the bit offset operand (second operand) and stores the value of the bit in the CF flag. The bit base operand can be a register or a memory location; the bit offset operand can be a register or an immediate value. If the bit base operand specifies a register, the instruction takes the modulo 16 or 32 (depending on the register size) of the bit offset operand, allowing any bit position to be selected in a 16- or 32-bit register, respectively (see Figure 38-1). If the bit base operand specifies a memory location, it represents the address of the byte in memory that contains the bit base (bit 0 of the specified byte) of the bit string (see Figure 38-2). The offset operand then selects a bit position within the range -2^{31} to $2^{31} - 1$ for a register offset and 0 to 31 for an immediate offset.

Some assemblers support immediate bit offsets larger than 31 by using the immediate bit offset field in combination with the displacement field of the memory operand. In this case, the low-order 3 or 5 bits (3 for 16-bit operands, 5 for 32-bit operands) of the immediate bit offset are stored in the immediate bit offset field, and the high-order bits are shifted and combined with the byte displacement in the addressing mode by the assembler. The processor will ignore the high order bits if they are not zero.

When accessing a bit in memory, the processor may access 4 bytes starting from the memory address for a 32-bit operand size, using by the following relationship:

$$\text{Effective Address} + (4 * (\text{BitOffset} \text{ DIV } 32))$$

Or, it may access 2 bytes starting from the memory address for a 16-bit operand, using this relationship:

$$\text{Effective Address} + (2 * (\text{BitOffset} \text{ DIV } 16))$$

It may do so even when only a single byte needs to be accessed to reach the given bit. When using this bit addressing mechanism, software should avoid referencing areas of memory close to address space holes. In particular, it should avoid references to memory-mapped I/O registers. Instead, software should use the MOV instructions to load from or store to these addresses, and use the register form of these instructions to manipulate the data.

Operation

$$CF \leftarrow \text{Bit}(\text{BitBase}, \text{BitOffset})$$

Flags Affected

The CF flag contains the value of the selected bit. The OF, SF, ZF, AF, and PF flags are undefined.

Protected Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.

#AC(0) If alignment checking is enabled and an unaligned memory reference is made.

40.6 BTC—Bit Test and Complement

Opcode	Instruction	Description
0F BB	BTC <i>r/m16,r16</i>	Store selected bit in CF flag and complement
0F BB	BTC <i>r/m32,r32</i>	Store selected bit in CF flag and complement
0F BA /7 <i>ib</i>	BTC <i>r/m16,imm8</i>	Store selected bit in CF flag and complement
0F BA /7 <i>ib</i>	BTC <i>r/m32,imm8</i>	Store selected bit in CF flag and complement

Description

Selects the bit in a bit string (specified with the first operand, called the bit base) at the bit-position designated by the bit offset operand (second operand), stores the value of the bit in the CF flag, and complements the selected bit in the bit string. The bit base operand can be a register or a memory location; the bit offset operand can be a register or an immediate value. If the bit base operand specifies a register, the instruction takes the modulo 16 or 32 (depending on the register size) of the bit offset operand, allowing any bit position to be selected in a 16- or 32-bit register, respectively (see Figure 38-1). If the bit base operand specifies a memory location, it represents the address of the byte in memory that contains the bit base (bit 0 of the specified byte) of the bit string (see Figure 38-2). The offset operand then selects a bit position within the range -2^{31} to $2^{31} - 1$ for a register offset and 0 to 31 for an immediate offset.

Some assemblers support immediate bit offsets larger than 31 by using the immediate bit offset field in combination with the displacement field of the memory operand. See “BT—Bit Test” in this chapter for more information on this addressing mechanism.

Operation

```
CF ← Bit(BitBase, BitOffset)
Bit(BitBase, BitOffset) ← NOT Bit(BitBase, BitOffset);
```

Flags Affected

The CF flag contains the value of the selected bit before it is complemented. The OF, SF, ZF, AF, and PF flags are undefined.

Protected Mode Exceptions

#GP(0)	If the destination operand points to a nonwritable segment.
	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
	If the DS, ES, FS, or GS register contains a null segment selector.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

- #GP If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
- #SS If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

- #GP(0) If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
- #SS(0) If a memory operand effective address is outside the SS segment limit.
- #PF(fault-code) If a page fault occurs.
- #AC(0) If alignment checking is enabled and an unaligned memory reference is made.

40.7 BTR—Bit Test and Reset

Opcode	Instruction	Description
0F B3	BTR <i>r/m16,r16</i>	Store selected bit in CF flag and clear
0F B3	BTR <i>r/m32,r32</i>	Store selected bit in CF flag and clear
0F BA /6 <i>ib</i>	BTR <i>r/m16,imm8</i>	Store selected bit in CF flag and clear
0F BA /6 <i>ib</i>	BTR <i>r/m32,imm8</i>	Store selected bit in CF flag and clear

Description

Selects the bit in a bit string (specified with the first operand, called the bit base) at the bit-position designated by the bit offset operand (second operand), stores the value of the bit in the CF flag, and clears the selected bit in the bit string to 0. The bit base operand can be a register or a memory location; the bit offset operand can be a register or an immediate value. If the bit base operand specifies a register, the instruction takes the modulo 16 or 32 (depending on the register size) of the bit offset operand, allowing any bit position to be selected in a 16- or 32-bit register, respectively (see Figure 38-1). If the bit base operand specifies a memory location, it represents the address of the byte in memory that contains the bit base (bit 0 of the specified byte) of the bit string (see Figure 38-2). The offset operand then selects a bit position within the range -2^{31} to $2^{31} - 1$ for a register offset and 0 to 31 for an immediate offset.

Some assemblers support immediate bit offsets larger than 31 by using the immediate bit offset field in combination with the displacement field of the memory operand. See “BT—Bit Test” in this chapter for more information on this addressing mechanism.

Operation

```
CF ← Bit(BitBase, BitOffset)
Bit(BitBase, BitOffset) ← 0;
```

Flags Affected

The CF flag contains the value of the selected bit before it is cleared. The OF, SF, ZF, AF, and PF flags are undefined.

Protected Mode Exceptions

#GP(0)	<p>If the destination operand points to a nonwritable segment.</p> <p>If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.</p> <p>If the DS, ES, FS, or GS register contains a null segment selector.</p>
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP(0)	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.

40.8 BTS—Bit Test and Set

Opcode	Instruction	Description
0F AB	BTS <i>r/m16,r16</i>	Store selected bit in CF flag and set
0F AB	BTS <i>r/m32,r32</i>	Store selected bit in CF flag and set
0F BA /5 <i>ib</i>	BTS <i>r/m16,imm8</i>	Store selected bit in CF flag and set
0F BA /5 <i>ib</i>	BTS <i>r/m32,imm8</i>	Store selected bit in CF flag and set

Description

Selects the bit in a bit string (specified with the first operand, called the bit base) at the bit-position designated by the bit offset operand (second operand), stores the value of the bit in the CF flag, and sets the selected bit in the bit string to 1. The bit base operand can be a register or a memory location; the bit offset operand can be a register or an immediate value. If the bit base operand specifies a register, the instruction takes the modulo 16 or 32 (depending on the register size) of the bit offset operand, allowing any bit position to be selected in a 16- or 32-bit register, respectively (see Figure 38-1). If the bit base operand specifies a memory location, it represents the address of the byte in memory that contains the bit base (bit 0 of the specified byte) of the bit string (see Figure 38-2). The offset operand then selects a bit position within the range -2^{31} to $2^{31} - 1$ for a register offset and 0 to 31 for an immediate offset.

Some assemblers support immediate bit offsets larger than 31 by using the immediate bit offset field in combination with the displacement field of the memory operand. See “BT—Bit Test” in this chapter for more information on this addressing mechanism.

Operation

```
CF ← Bit(BitBase, BitOffset)
Bit(BitBase, BitOffset) ← 1;
```

Flags Affected

The CF flag contains the value of the selected bit before it is set. The OF, SF, ZF, AF, and PF flags are undefined.

Protected Mode Exceptions

#GP(0)	<p>If the destination operand points to a nonwritable segment.</p> <p>If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.</p> <p>If the DS, ES, FS, or GS register contains a null segment selector.</p>
#SS(0)	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made while the current privilege level is 3.

Real-Address Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.

Virtual-8086 Mode Exceptions

#GP	If a memory operand effective address is outside the CS, DS, ES, FS, or GS segment limit.
#SS	If a memory operand effective address is outside the SS segment limit.
#PF(fault-code)	If a page fault occurs.
#AC(0)	If alignment checking is enabled and an unaligned memory reference is made.